Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

1. (Previously presented) A method comprising:

converting memory access instructions in a source code into intermediary standard formatted memory access instructions;

generating a plurality of memory access partitions containing corresponding subsets of the intermediary standard formatted memory access instructions, with the plurality of memory access partitions directed to specific memory banks;

identifying matching instructions based on comparisons of pre-defined instruction patterns to the intermediary standard formatted memory access instructions in the plurality of memory access partitions; and

transforming the identified matching instructions to vector memory access instructions, with the transformed vector memory access instructions, when executed, causing a corresponding memory access operation to be performed for a group of memory locations.

2. (Original) The method of claim 1 in which converting comprises converting memory access instructions that read or write less than a minimum data access unit (MDAU) to memory access instructions that read or write a multiple of the minimum data access unit.

3. (Original) The method of claim 2 in which converting further comprises transforming the memory access instructions that read or write the multiple of the minimum data access unit to a format including a base address plus an offset.

4. (Previously presented) The method of claim 1 in which generating the plurality of memory access partitions comprises:

generating a data flow graph containing basic blocks including the intermediary standard formatted memory access instructions; and

for each basic block, applying a set of rules.

5. (Cancelled)

6. (Previously presented) The method of claim 4 in which applying comprises limiting a subnode of one of the plurality of memory access partitions to a memory read or a memory write.

7. (Previously presented) The method of claim 1 in which the memory banks include a static random access memory (SRAM).

8. (Previously presented) The method of claim 1 in which the memory banks include a dynamic random access memory (DRAM).

9. (Previously presented) The method of claim 1 in which the memory banks include a scratchpad memory.

10. (Previously presented) The method of claim 1 in which the memory banks include an EEPROM.

11. (Previously presented) The method of claim 1 in which the memory banks include flash memory.

12. (Previously presented) The method of claim 1 in which the memory banks include a NVRAM.

13. (Original) The method of claim 1 in which the instruction patterns comprise a pattern describing instruction semantics.

14. (Original) The method of claim 1 in which the vector memory access instructions comprise single memory access instructions representing multiple memory accesses to a type of memory.

15. (Previously presented) A compilation method comprising:

converting source code that includes memory access instructions that read or write less than a minimum data access unit (MDAU) to intermediary code that includes memory access instructions that read or write a multiple of the minimum data access unit;

converting the memory access instructions of the intermediary code into intermediary memory access instructions that have a format including a base address plus an offset;

grouping subsets of the intermediary memory access instructions into a plurality of memory access partitions, with the plurality of memory access partitions containing intermediate memory access instructions directed to specific memory banks; and

transforming the intermediary memory access instructions in the subsets corresponding to the plurality of memory access partitions that match pre-defined instruction patterns to vector memory access instructions, with the transformed vector memory access instructions, when executed, causing a corresponding memory access operation to be performed for a group of memory locations.

16. (Previously presented) The compilation method of claim 15 in which grouping comprises:

generating a data flow graph containing basic blocks including intermediary memory access instructions; and

generating subnodes in the plurality of the memory access partitions, each subnode including intermediary memory access instructions directed to the same operation in a memory bank corresponding to the respective memory access partition.

17. (Original) The compilation method of claim 16 in which the operation is a read.

18. (Original) The compilation method of claim 16 in which the operation is a write.

19. (Original) The compilation method of claim 16 in which the memory bank is a static random access memory (SRAM).

20. (Original) The compilation method of claim 16 in which the memory bank is a dynamic random access memory (DRAM).

21. (Original) The compilation method of claim 16 in which the memory bank is a scratchpad memory.

22. (Original) The compilation method of claim 16 in which the memory bank is an EEPROM.

23. (Original) The compilation method of claim 16 in which the memory bank is flash memory.

24. (Original) The compilation method of claim 16 in which the memory bank is NVRAM.

25. (Original) The compilation method of claim 15 in which the instruction patterns comprises instruction semantics.

26. (Original) The compilation method of claim 25 in which the instruction semantics comprises segments.

27. (Currently amended) A computer program product, for vectorizing memory access instructions, the computer program product residing on a computer ~~machine~~-readable medium for storing computer instructions that, when executed, cause data processing apparatus to:

convert memory access instructions residing in a source code into intermediary standard formatted memory access instructions;

generate a plurality of memory access partitions containing corresponding subsets of the intermediary standard formatted memory access instructions, with the plurality of memory access partitions directed to specific memory banks;

identify matching instruction based on comparisons of pre-defined instruction patterns to the intermediary standard formatted memory access instructions in the plurality of memory access partitions; and

transform the identified matching instructions to vector memory access instructions, with the transformed vector memory access instructions, when executed, causing a corresponding memory access operation to be performed for a group of memory locations.

28. (Previously presented) The computer program product of claim 27, the computer instruction that cause the data processing apparatus to convert comprise computer instructions that cause the data processing apparatus to convert memory access instructions that read or write less than a minimum data access unit to memory access instructions that read or write a multiple of the minimum data access unit.

29. (Previously presented) The computer program product of claim 28, the computer instruction that cause the data processing apparatus to convert memory access instructions further comprise computer instructions that cause the data processing apparatus to transform the

memory access instructions that read or write the multiple of the minimum data access unit to a format including a base address plus an offset.

30. (Previously presented) The computer program product of claim 27, the computer instruction that cause the data processing apparatus to generate partitions comprise computer instructions that cause the data processing apparatus to:

generate a data flow graph containing basic blocks including the intermediary standard formatted memory access instructions; and

generate subnodes in the plurality of memory access partitions, the subnodes including intermediary standard formatted memory access instructions directed to the same operation in the memory banks corresponding to the respective memory access partitions.